INTERNATIONAL STANDARD

ISO/IEC 30130

First edition

Software engineering — Capabilities of software testing tools

Ingénierie du logiciel — Capacités des outils d'essai de logiciel

PROOF/ÉPREUVE

Reference number
ISO/IEC 30130:2015(E)

Contents

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular the different approval criteria needed for the different types of document should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation on the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the WTO principles in the Technical Barriers to Trade (TBT) see the following URL: Foreword - Supplementary information

The committee responsible for this document is ISO/IEC JTC 1, *Information technology*, SC 7, *Software and systems engineering*.

Introduction

This International Standard defines the framework to which capabilities of software testing tools are allocated in order to identify the capabilities of products being used by any project for software testing. To develop high-quality software with reasonable time and budget, the use of software testing tools is required. The increase in the size and complexity of software is complemented by an increase in the difficulty and complexity of software testing. This created a larger demand for the support of tools in order to test software effectively and efficiently.

Testing tools are highly diverse due to their contexts of use. Testing itself varies by objective, such as functional testing or nonfunctional testing, and granularity, such as unit testing or system testing. Testing tool vendors vary by providing tools with a different function or combination of functions. And despite vendor provided explanations for the type of testing support functions, there is little common understanding of these functions. In this environment, it is difficult to utilize a testing tool that is suitable for a project without common understanding of tool functions, proper acquisition of the needed tools, and efficient training.

The framework defined by this International Standard consists of objectives of testing, granularity of software to be tested and capabilities. In Clause 4, an object model for software testing tools as basis for the framework is defined. In Clause 5, three of the categories in that software testing model (Dynamic Test Execution, Code Analysis, and Test Management) are specified. In Clause 6, quality characteristics, granularity, and other aspects of characteristics are defined and in Clause 7, tool capabilities are mapped onto those categories and characteristics.

Software engineering — Capabilities of software testing tools

1 Scope

This International Standard defines the framework to which capabilities of software testing tools are allocated in order to identify the capabilities of products being used by any project for software testing. Software testing processes are identified in ISO/IEC/IEEE 29119-2 and software verification processes are identified in ISO/IEC 12207. This International Standard is fully harmonized with these existing standards in terms of software testing processes.

This International Standard focuses on the following areas that the existing ISO/IEC standards do not deal with the following:

— categorization of software test entities and software testing tools (Clauses 4 and 5);

— characterization of each software testing tool category (Clauses 5 and 6);

— mapping of software testing tool capabilities and characteristics (Clauses 6 and 7).

2 Normative references

The following documents, in whole or in part, are normatively referenced in this document and are indispensable for its application. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO/IEC 25010, *Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models*

3 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

3.1
quality record report
report which is generated through dynamic test execution and code analysis to record test results and other output

Note 1 to entry: Including Test Result, Static Code Analysis Report, Test Incident Report, and Metrics.

3.2
test target version
specific version of test target which is used for one-time execution of Dynamic Test Execution or Code Analysis

3.3
testing tool
specific or generic tool which is used for test execution and test management such as test results recording, test results display, test results interpretation, generation of test data, generation of test procedure, generation of test scripts, test modelling, etc.

3.4
package
namespace for the grouped elements

4 Object model for software testing tools

4.1 Overview of the object model

A software testing tool is described by its input, process, output and environment. To define software testing tools, the object model for software testing is identified.

The object model for software testing tools consists of the following elements:

a) test process, which represents processes of dynamic test;

b) code analysis process, which represents processes of code analysis;

c) test entity, which represents entities that appear in the process;

d) test tool, which supports inputs, processes, outputs and test entity.

Test Process comprises multiple subprocesses. Each process has input and output. These input and output are basically test entities. Test processes are specified in detail by ISO/IEC/IEEE 29119-2.

Code Analysis Process comprises multiple sub processes. Each process has input and output. These input and output are basically test entities.

Test Entity comprises multiple entities. The entity is referred to as Dynamic Test Execution Entity that is required or created in Dynamic Test Process. The entity is referred to as Code Analysis Entity that is required or created in Code Analysis Process. A test entity represents a single performance of a test, in which the target of the test is tested in various respects. Testing has two modes of execution. Dynamic Test Execution involves actual execution of the code and Code Analysis involves examination of the source code of the target.

The testing tools take test entities as input and produce test entities. By producing test entities, testing tools effectively support the testing process.

The object model diagrams, Figures 1 to 5, are described using UML 2 (ISO/IEC 19505-2).

Test Process includes Dynamic Test Process and Test Management Process.

Code Analysis Process includes Code Analysis Process and Code Analysis Management Process.

Test Entity includes Test Management Entity, Dynamic Test Execution Entity and Code Analysis Entity, and Test Target.

Test Tool includes Test Execution Tool and Test Management Tool.

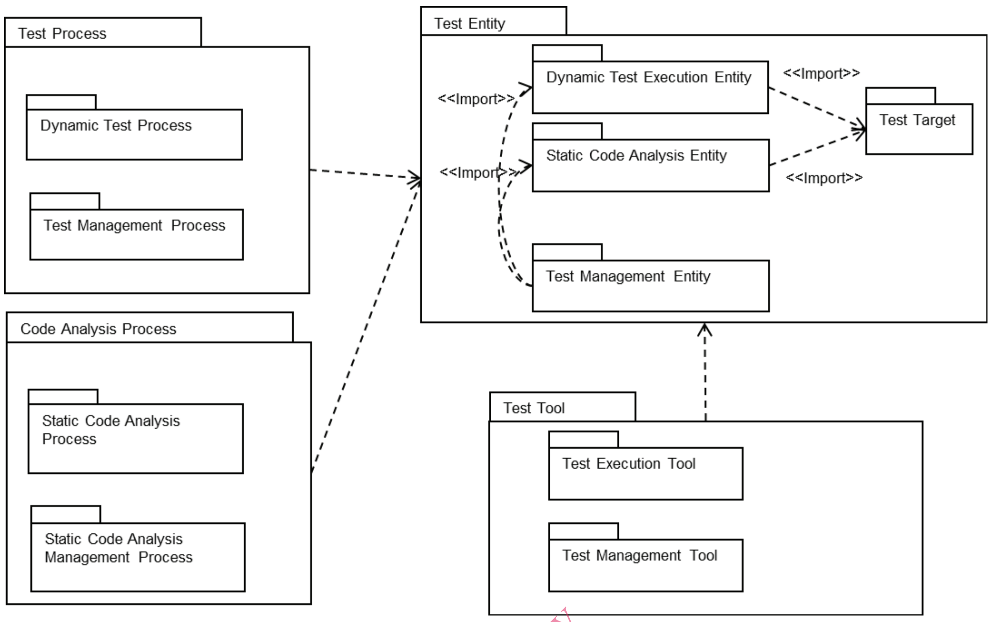

Figure 1 — Object model of software testing

Test Process and Test Tool have indirect relationships which are described in Annex A.

4.2 Test target

Test Target is a set of source code and executable code of software that is in development. It has a hierarchical structure, and any part of the structure or the whole structure is tested.

Test Target Version is a specific version of the software.

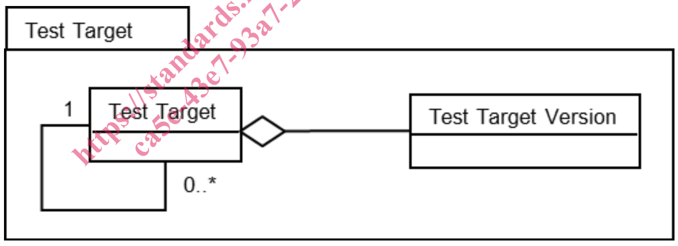

Figure 2 — Object model of the package "Test Target"

4.3 Dynamic test execution entity

Dynamic Test Execution Entity is prepared to identify necessary entities which are used in one-time dynamic test execution. Test Data, Test Target Version, Test Result, and Expected Result are identified in this package.

In one-time dynamic test execution, Test Environment is used.

Test Target Version which is used in one-time dynamic test execution is extracted from an appropriate granularity and version of Test Target.

Test Data which is used in one-time dynamic test execution is designed for Test Case which is derived from Specification and is stored on Test Data Collections in Test Data Repository.

Specification, Test Case, and Test Target are referred to as input for Dynamic Test Execution.

Test Data Repository has Test Data which is used by all Dynamic Test Execution and is classified by Objective and Technique.

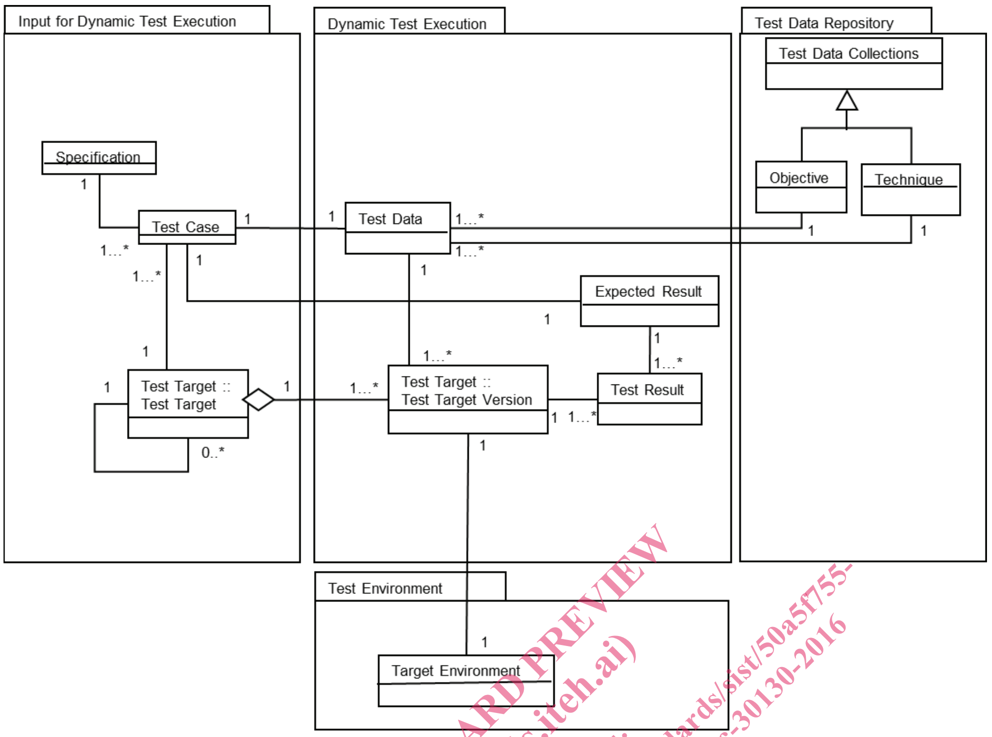

Figure 3 — Object model of the package “Dynamic Test Execution Entity”

4.4 Code analysis entity

Code Analysis Entity is prepared to identify necessary entities which are used in one-time code analysis. Test Target Version and Code Analysis Result are identified in this package.

Test Target Version which is used in one-time code analysis is extracted from an appropriate granularity and version of Test Target.

Test Target and Check List which are used in Code Analysis are referred to as Input for Code Analysis. Check List usually does not depend on Specification.

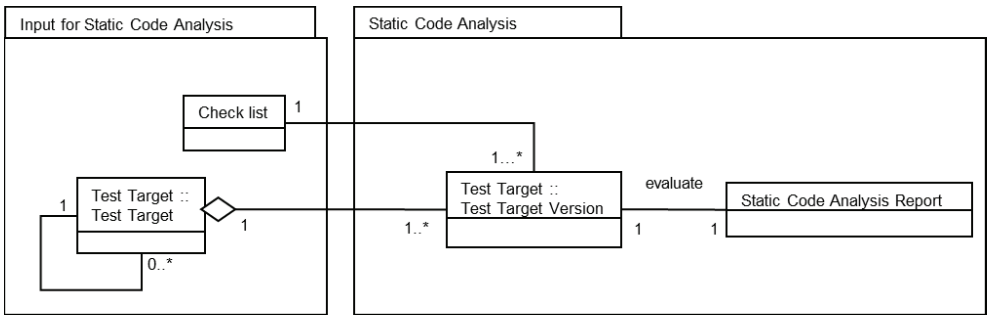
Input for Static Code Analysis
Static Code Analysis
Check list
Test Target :: Test Target
Test Target :: Test Target Version
evaluate
Static Code Analysis Report
1
1…*
1..*
0..*

Figure 4 — Object model of the package “Code Analysis Entity”

4.5 Test management entity

Test Management Entity is prepared to identify necessary packages and entities which are used to manage the entire test processes. Test Plan, Test Asset, Quality Record Report, Test Completion Report ,Verification and Validation Report,and Test Status Report are identified in this package.

Dynamic Test Execution Documentation and Static Code Analysis Report are referred to as Quality Record Report.

Dynamic Test Execution Documentation is composed by Metric, Test Result, and Test Incident Report. Static Code Analysis Report is composed by Metric, Static Code Analysis Report, and Test Incident Report.

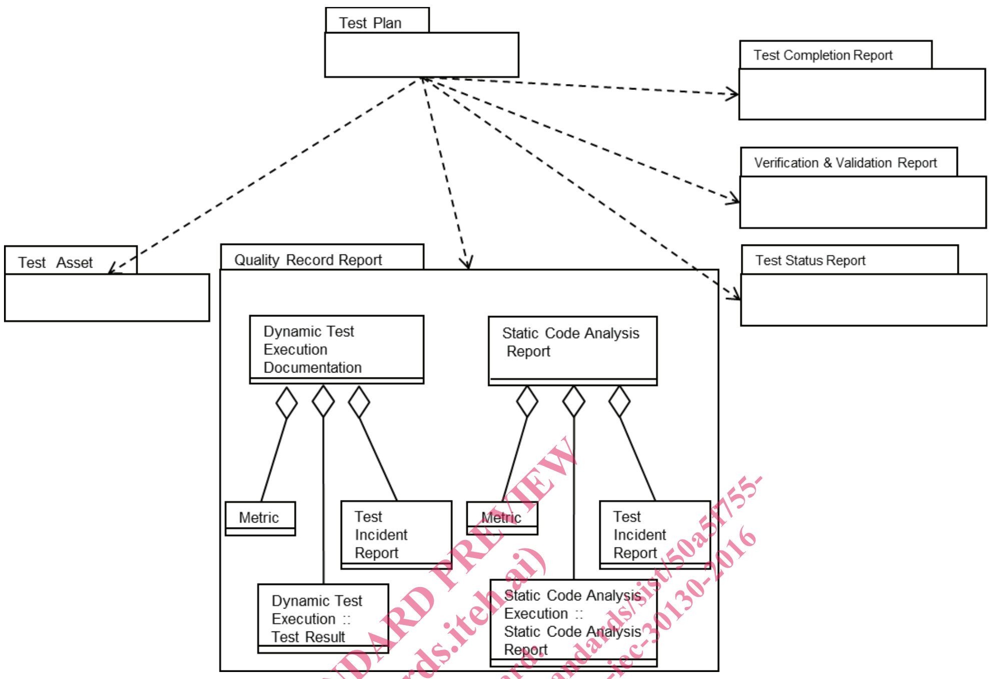
Test Plan
Test Completion Report
Verification & Validation Report
Test Status Report
Test Asset
Quality Record Report
Dynamic Test Execution Documentation
Static Code Analysis Report
Metric
Test Incident Report
Metric
Test Incident Report
Dynamic Test Execution :: Test Result
Static Code Analysis Execution :: Static Code Analysis Report

Figure 5 — Object model of the package “Test Management Entity”

5 Category of test entity

5.1 Overview

In this Clause, categories of test entities are defined to allocate and specify the capabilities of software tools. Each category is identified by analyzing objective, life history, and usage of each test entity.

5.2 Categories of dynamic test execution entities

5.2.1 Input for dynamic test execution

Entities in this category are used in Dynamic Test Execution as inputs, such as Specification, Test Case, and Test Target.

5.2.2 Dynamic test execution

Entities in this category are used in one-time dynamic testing. It comprises Test Data, Test Result, Expected Result, and Test Target Version.

5.2.3 Test data repository

Entities in this category are used in Dynamic Test Execution. It comprises Test Data Collections, in which there are a large number of test data for the overall Dynamic Test Execution. Test Data Collections can be subsets which depend on each objective and techniques.